

## Task A22. Cycles

 1 s  256 MB

Yan este un băiat curajos, care alege întotdeauna drumul cel mai scurt de la școală până la casa bunicii. Din păcate, acest drum trece prin pădurea bântuită.

Într-o zi, mergând prin pădure, Yan întâlnește creatura înfricoșătoare Tung Tung Sahur. Această creatură îl provoacă la un joc: dacă Yan câștigă, atunci va putea să își continue drumul – și ar putea chiar să primească o recompensă. Totuși, dacă pierde, Tung Tung Sahur îl va mânca.

Jocul începe cu Tung Tung Sahur alegând o permutare ascunsă  $P$  cu  $N$  elemente. Acum, este rândul lui Yan să joace. El poate alege de mai multe ori două poziții diferite din permutare, după care îi va cere lui Tung Tung Sahur să interschimbe elementele de la acele poziții. Tung Tung Sahur va face interschimbarea și va spune numărul de **cicluri**<sup>1</sup> din permutarea actualizată. Aceste interschimbări sunt persistente – Tung Tung Sahur nu le anulează (dar Yan este liber să repete o interschimbare pentru a o inversa dacă dorește).

În orice moment, Yan poate declara că permutarea este sortată strigând “Sorted.” Dacă permutarea chiar este sortată crescător, Tung Tung Sahur îl va lăsa să plece. În plus, Tung Tung Sahur îl va recompensa pe Yan cu monede de aur, în funcție de cât de puține interschimbări au fost necesare. Așadar, Yan trebuie să sorteze permutarea folosind cât mai puține interschimbări.

Sarcina voastră este să-l ajutați pe Yan să sorteze permutarea ascunsă folosind doar informația dată și să faceți asta minimizând numărul de interschimbări necesare.

### Detalii de implementare

Concurentul are de implementat funcția:

```
void sortPermutation(int N);
```

Aceasta va fi apelată o singură dată pentru fiecare test, cu  $N$  — numărul de elemente din permutarea pe care Yan trebuie să o sorteze. Din această funcție (și alte funcții pe care le scrieți), puteți apela funcția `performSwap` pentru a interschimba două elemente din permutare și veți primi numărul de cicluri din permutarea rezultată. Trebuie să garantați că, la finalul funcției, permutarea ascunsă este sortată crescător, adică  $P_i = i$  pentru orice  $0 \leq i < N$ , deoarece Yan va striga imediat “Sorted” după ce funcția se termină.

```
int performSwap(int x, int y)
```

Apelul `performSwap` reprezintă interschimbarea elementului  $P_x$  cu  $P_y$  (pentru  $0 \leq x, y \leq N - 1$ ). Atenție, dacă apeleți această funcție cu același element, adică  $x = y$ , atunci veți primi Wrong Answer. Această funcție returnează numărul de cicluri din permutarea **actualizată**.

<sup>1</sup>Orice permutare a numerelor întregi  $\{0, 1, \dots, N - 1\}$  poate fi descompusă într-o mulțime de cicluri disjuncte. Pentru a găsi un astfel de ciclu, pornim de la orice indice  $i$ , și urmăm maparea  $i \mapsto P_i \mapsto P_{P_i} \mapsto \dots$  până ne întoarcem la  $i$  – acestea alcătuiesc un ciclu. De exemplu, permutarea  $[1, 2, 0, 5, 4, 3]$  reprezintă maparea  $0 \mapsto 1, 1 \mapsto 2, 2 \mapsto 0, 3 \mapsto 5, 4 \mapsto 4, 5 \mapsto 3$  și este alcătuită din ciclurile disjuncte  $(0\ 1\ 2)$ ,  $(3\ 5)$  și  $(4)$ , deci are trei cicluri.

### Testare locală

Pentru testarea locală a programului, un evaluator local și un fișier header sunt prevăzute. Evaluatorul local citește mai întâi  $N$  și apoi  $N$  numere distincte de la 0 la  $N - 1$ . Apoi, apelează funcția `sortPermutation` implementată de voi și se așteaptă ca permutarea să fie sortată corect.

### Restricții

- $N = 1000$

### Subtask-uri

Subtask	Punctaj	Restricții	Restricții suplimentare
1	10	$N = 1000$	Pentru $i$ par: $(P_i, P_{i+1}) = (i, i + 1)$ or $(i + 1, i)$ .
2	20	$N = 1000$	Permutarea inițială poate fi sortată dintr-o singură interschimbare.
3	70	$N = 1000$	–

Se acordă o fracțiune din punctaj pentru un subtask doar dacă soluția trece cu succes toate testele din el. Rezultatul va fi calculat în funcție de  $Q$  – numărul maxim de apeluri ale funcției `performSwap` în cadrul tuturor testelor:

$10^7 < Q$	0%
$10^6 < Q \leq 10^7$	10%
$9 \cdot 10^4 < Q \leq 10^6$	20%
$3 \cdot 10^4 < Q \leq 9 \cdot 10^4$	60%
$Q \leq 3 \cdot 10^4$	100%

### Exemplu

Input	Interacțiune
3 2 0 1	<code>sortPermutation(3)</code> <code>performSwap(0, 1): return 2</code> <code>performSwap(0, 1): return 1</code> <code>performSwap(0, 1): return 2</code> <code>performSwap(1, 2): return 3</code> <code>sortPermutation(3): returns</code>